IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received April 2, 2022, accepted April 12, 2022, date of publication April 18, 2022, date of current version April 28, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3168665

Fast Edit Propagation for 360 Degree Panoramas Using Function Interpolation

YUN ZHANG[1], FANG-LUE ZHANG[2], (Member, IEEE), ZHE ZHU[3], LIDONG WANG[4], AND YAO JIN[5]

[1]College of Media Engineering, Communication University of Zhejiang, Hangzhou 310018, China
[2]School of Engineering and Computer Science, Victoria University of Wellington, Wellington 6012, New Zealand
[3]Mathworks, Natick, MA 01760, USA
[4]Qianjiang College, Hangzhou Normal University, Hangzhou 310018, China
[5]School of Informatics Science and Technology, Zhejiang Sci-Tech University, Hangzhou 310018, China

Corresponding author: Yun Zhang (zhangyun@cuz.edu.cn)

This work was supported in part by the Zhejiang Province Public Welfare Technology Application Research under Grant LGG22F020009, in part by the Key Laboratory of Film and TV Media Technology of Zhejiang Province under Grant 2020E10015, in part by the Teaching Reform Project of Communication University of Zhejiang under Grant jgxm202131, in part by the Zhejiang Provincial Natural Science Foundation of China under Grant LY19F020022, and in part by the Marsden Fund Council managed by the Royal Society of New Zealand under Grant MFP-20-VUW-180.

ABSTRACT Due to its intuitive stroke-based user interface, edit propagation has wide applications such as image recoloring and relighting, tone editing, image matting, etc. This paper presents a function interpolation method for efficient edit propagation on 360 degree panoramas. Given a panoramic image as input and user-specified strokes as edit hints, our approach adopts an adaptive sampling strategy based on the diversity of local color distribution. By selecting the most representative samples, this strategy can significantly reduce the number of samples. We assume that the projection of the input panorama is known, so that it can be mapped to a 3D unit sphere where the spherical distance metric can be defined. Then edit propagation is formulated as a radial basis function interpolation problem, which can be efficiently solved by using the non-negative linear least squares. We also utilize a multi-resolution strategy and a function look-up table for further acceleration. Experimental results show that our method can efficiently propagate edits on high-resolution panoramas and produce seam-free and smooth editing results.

INDEX TERMS Edit propagation, 360 degree panorama, function interpolation, adaptive sampling, seam-free.

I. INTRODUCTION

Metaverse is the most fashionable concept on the Internet today. As major technology companies announce their entry into this field, Virtual Reality (VR) technology is gaining more and more attention recently. As one of the key techniques, VR content manipulation, such as 360° panoramic image and video editing, has attracted much research attention and many efforts have been made to improve the effectiveness and efficiency of the editing tools. Given the massive demand, it has been increasingly important to create an intuitive and efficient editing tool for VR content manipulation.

Panoramas are produced by stitching multiple images with partial overlapping [1], [2], so they usually contain much more pixels than ordinary images, and require more efforts in editing and processing. To alleviate the labor-intensive region selection in image appearance (e.g., color, brightness) editing, the stroke-based interaction has been proposed [3]–[5]. Instead of precisely selecting all the pixels for editing, stroke-based tools allow users to draw only a few strokes in the regions of interest, where the edits can be propagated to related regions instantly. We state that stroke-based editing is ideal for panorama editing because of its intuitive interaction and efficiency. However, due to their spherical nature, applying the method directly can not ensure satisfactory results, and may lead to unexpected seams and inconsistent propagation, see Fig. 1.

Very recently, Zhang *et al.* [6] proposed an efficient manifold preserving edit propagation on 360° panoramas. To obtain seam-free and visually pleasing editing results, they

The associate editor coordinating the review of this manuscript and approving it for publication was Hossein Rahmani.

introduced the spherical distance metric and constructed the manifold structure for each pixel in the spherical domain. Despite the successful applications on many real world examples, the method in [6] still suffers from several problems. Firstly, their spherical distance metric is approximated by the chord length between two points on a sphere, which is not that accurate under certain circumstances. Secondly, it is still time-consuming to find the K-nearest neighbors even with the multi-resolution strategy. Thirdly, the algorithm is composed of several components, including K-D tree construction, KNN search, and K-D tree interpolation, which are complex and not easy to implement.

In previous works, the most efficient and easy-to-implement edit propagation approach is proposed by Li *et al.* [7], who formulated the propagation as a function interpolation problem. The complexity of their method is independent of the resolution but closely related with the samples on strokes; thus, it is speedy for sparse edits. However, for 360° panoramas, the number of samples is always significant due to their high-resolution nature, which may increase the computational cost.

Inspired by previous works [6], [7], we propose a function interpolation method for edit propagation on 360° panoramas. Our key observation is that strokes in complex regions require more samples to construct the smooth functions in the affinity space. Thus, instead of randomly sampling a certain proportion of samples, we adaptively determine the number of samples on each stroke. Secondly, for seam-free and consistent propagation, we incorporate the spherical distance metric into the panorama pixel distance calculation. Given a panorama with known projection as input, we first map the planar image to a unit sphere. The distance between pixels can be calculated by the considerable circle distance on a sphere. We formulate the edit propagation as a radial basis function interpolation problem, which is easy to implement, and only needs to solve a small linear system whose size is proportional to the number of samples. We further accelerate the algorithm by a multi-resolution strategy and a function look-up table.

Actually, compared with the manifold preserving propagation [8]–[10], the function interpolation based method may cause halo artifacts at regions with color blending. However, for edit propagation on 360° panoramas, we focus on changing the appearance of large-scale background and foreground, e.g., sky, grassland, and floor. Thus, the color blending problem in edit propagation can always be ignored. Our method can reach a good balance between efficiency and visual effects, and the main contributions can be listed as follows:

- We propose a function interpolation based method for edit propagation on 360° panoramas, which can produce seam-free and consistent propagation in the spherical domain.
- For efficient propagation on 360° panoramas with tens of millions of pixels, we propose an adaptive sampling scheme for function interpolation, which can greatly reduce the computational cost without sacrificing the editing quality by adaptively determining the number of samples according to the diversity of local color distribution.

The organization of this article is as follows. We first briefly review the techniques related to our work in Section II. Then we describe the detailed algorithm of edit propagation on sphere as well as the acceleration scheme in Section III, and evaluate the performance of our method in Section IV. Finally, we conclude the paper in Section V.

II. RELATED WORK

A. EDIT PROPAGATION

Edit propagation aims to edit the appearance (e.g., color, brightness, contrast, etc.) of images/videos by propagating the sparse edits from user-specified strokes to other regions, which avoids the tedious and laborious interactions in region selection. Colorization using optimization [3] is a pioneering work in stroke-based image editing, which avoids precise image segmentation and accurate region tracking. Followed by [3], Lischinski *et al.* [4] presented interactive local tonal value adjustment by drawing sparse strokes and using sliders, which provides more control than previous methods. In 2008, An *et al.* put forward the term *edit propagation* for the first time in Appprop [5], where user-specified edits on strokes are automatically propagated to spatially-close regions with similar appearance. Although providing intuitive interactions, the propagation methods above often suffer from the huge memory and computation cost. To solve this problem, Xu *et al.* [11] proposed a K-D tree clustering-based approximation scheme for affinity-based edit propagation, which accelerates the calculation and reduces the memory cost without sacrificing the visual fidelity. Sampling-based methods have been widely used in image editing, such as matting [12]–[14], edit propagation [7], [15], due to the advantages in terms of speed and space. Huang *et al.* [14] proposed a pixel-level discrete multi-objective sampling method for matting, which aims to solve the conflicts among multiple sampling criteria and incomplete sample spaces problem. Li *et al.* [7] further reduced the time and space complexity of edit propagation by reformulating the propagation as a function interpolation problem in high dimensional space. Bie *et al.* [15] applied efficient stroke sampling to calculate the affinity between image pixels and strokes, which can significantly reduce the number of clusters, thus making great acceleration. Similar to [7], [15], Yatagawa *et al.* [16] proposed the sparse pixel sampling, and a new approximation model for image color and edit interpolation based on affine combinations. To improve the visual quality of edit propagation, Ma *et al.* [17] introduced antialias map to remove aliasing artifacts. Chen *et al.* [8] proposed manifold preserving edit propagation, which seeks to represent each pixel as a linear combination of its neighbors in high dimensional feature space by using locally linear embedding, and produces more robust and visually satisfactory results. Despite the advantages of [8], it has high memory and computational costs. For efficient manifold preserving edit

propagation, Ma *et al.* [9] and Chen *et al.* [10] proposed to solve the optimization on clusters instead of pixels by using the K-D tree and quadtree structures. To further improve the performance of edit propagation, many learning-based methods are proposed. Under the observation that similar features get analogous edits, Oh *et al.* [18] constructed edit propagation as a classification problem using the Supported Vector Machine in feature space. Chen *et al.* [19] proposed an efficient edit propagation for extensive data based on sparse and editable dictionary learning. Using the deep learning architecture, Endo *et al.* proposed DeepProp [20], which can extract deep features from a single image for edit propagation. Gui *et al.* [21] regarded edit propagation as a multi-class classification problem and built an end-to-end DNN to solve it.

B. 360° PANORAMA PROCESSING

Recently, 360° panoramas, which refer to 360° videos/images that cover a 360° × 180° viewing range, have become more and more popular due to the immersive experiences. Therefore, it has become more critical to process 360° panoramas efficiently. Xu *et al.* [22] conducted an extensive survey on state-of-the-art works in 360° video/image processing in terms of perception, assessment, and compression, and looked forward to the future research trends on it. At the same time, Wang *et al.* [23] surveyed the recent advances in VR content creation and exploration based on the deep learning framework, which includes the problems, future directions, and emerging research areas. They believed that the deep learning method in VR constitutes an emerging research area in visual media computing. Currently, researchers focus on processing for better viewing and navigation. Under the Atlanta world assumption, Jung *et al.* [24] proposed an optimization-based method for the upright adjustment of 360° spherical panoramas. Without the lines or horizon detection, a deep learning-based method [25] for automatic upright adjustment of 360° panoramas was proposed. To remove fast jitters in 360° videos and provide a better viewing experience, Koph [26] proposed a hybrid 2D-3D method, which applies a deformable rotation motion model to stabilize 360° videos. To cope with undesirable camera shaking and rotations, Tang *et al.* [27] proposed the joint stabilization and direction of 360° videos. For comfortable 360° video playback, Kang *et al.* [28] proposed an interactive and automatic 360° video navigation system, which can provide an optimized camera path that contains salient events. Although successful, [28] calculated only a single camera path for navigation. To catch all regions of interest in 360° video navigation, Wang *et al.* [29] presented Transitioning360, which can compute diverse content-aware NFoV virtual camera paths in a coarse-to-fine manner. For an optimal viewing experience, Lai *et al.* [30] presented a system for converting 360° videos into an NFOV hyper-lapse, using the visual saliency and scene semantics. Content editing is another direction for 360° panorama processing, but there is little research work in this field. Zhu *et al.* [31] proposed a novel method to complete holes in 360° street views panoramas by using the optimization-based projection, which effectively avoids distortions. Zhao *et al.* [32] addressed the cloning problem in 360° panoramas using the spherical mean value coordinates (SMVC) as weights. Sumantri *et al.* [33] proposed a hierarchical generative network for high quality 360° panorama synthesis. Very recently, Zhang *et al.* [6] proposed an efficient method to propagate sparse edits on 360° panoramic images using the manifold preserving framework, which is further accelerated by adaptive K-D tree clustering.

III. ALGORITHM

Fig. 1 shows the flow chart of our algorithm. The input to our method is an equirectangular panoramic image and some strokes indicating regions to relight/recolor (white/colored) or remain unchanged (black). We first quantize the input image using fewer color levels and adaptively collect samples on strokes according to the distribution of color clusters. Then, we construct Radial Basis Functions (RBF) on the sphere and obtain the edit at each pixel by RBF interpolation. Finally, the editing result is rendered according to the calculated edits.

We list notations frequently used in this paper as follows. Let P be the input equirectangular image, which is quantized into P' for adaptive sampling. User-specified strokes are defined as $\{\gamma_u\}$, and the color clusters on each stroke is represented as $\Omega_u = \{\Omega_{u,v}\}$, where v refers to the index of each cluster on the u^{th} stroke. To uniformly distribute the samples on strokes, we further define the pixel set in each cluster as $Q_{u,v} = \{Q^m_{u,v}\}$, where m is the index of each pixel.

A. FUNCTION INTERPOLATION BASED EDIT PROPAGATION

As stated in Section II-A, edit propagation aims to propagate the sparse edits on user-specified strokes to the entire image or video, where similar editing results appear on similar pixels. We characterize each pixel by a high dimensional feature vector $\mathbf{f}_i = \{\mathbf{c}_i/\sigma_c, \mathbf{p}_i/\sigma_p\}$, where $\mathbf{c}_i$ and $\mathbf{p}_i$ refer to the color (e.g., RGB or Lab space) and position (e.g., x and y coordinates) respectively, and σ_c and σ_p are used to control the importance of the two terms. To reduce the computation and memory cost in the optimization-based propagation, Li et al [7] reformulated the propagation as a function interpolation problem and solved it by using the Radial Basis Functions (RBF). In their method, the edit at each pixel $\{e_i | e_i \in [0, 1]\}$, which indicates the amount of change in color, light, etc., can be approximated by the summations of RBF, as follows:

$$e_i = \sum_{j \in \Psi} a_j \sigma(\|\mathbf{f}_i - \mathbf{f}_j\|), \tag{1}$$

where Ψ refers to the sample set randomly selected from the strokes; σ is the basis function of RBF, which is defined as Gaussian function, i.e., $\sigma(x) = e^{-x^2}$; $\mathbf{f}_i$ is the feature vector of a pixel as described above. To obtain the coefficients $\{a_j\}$, we need to solve the following equation, which aims to

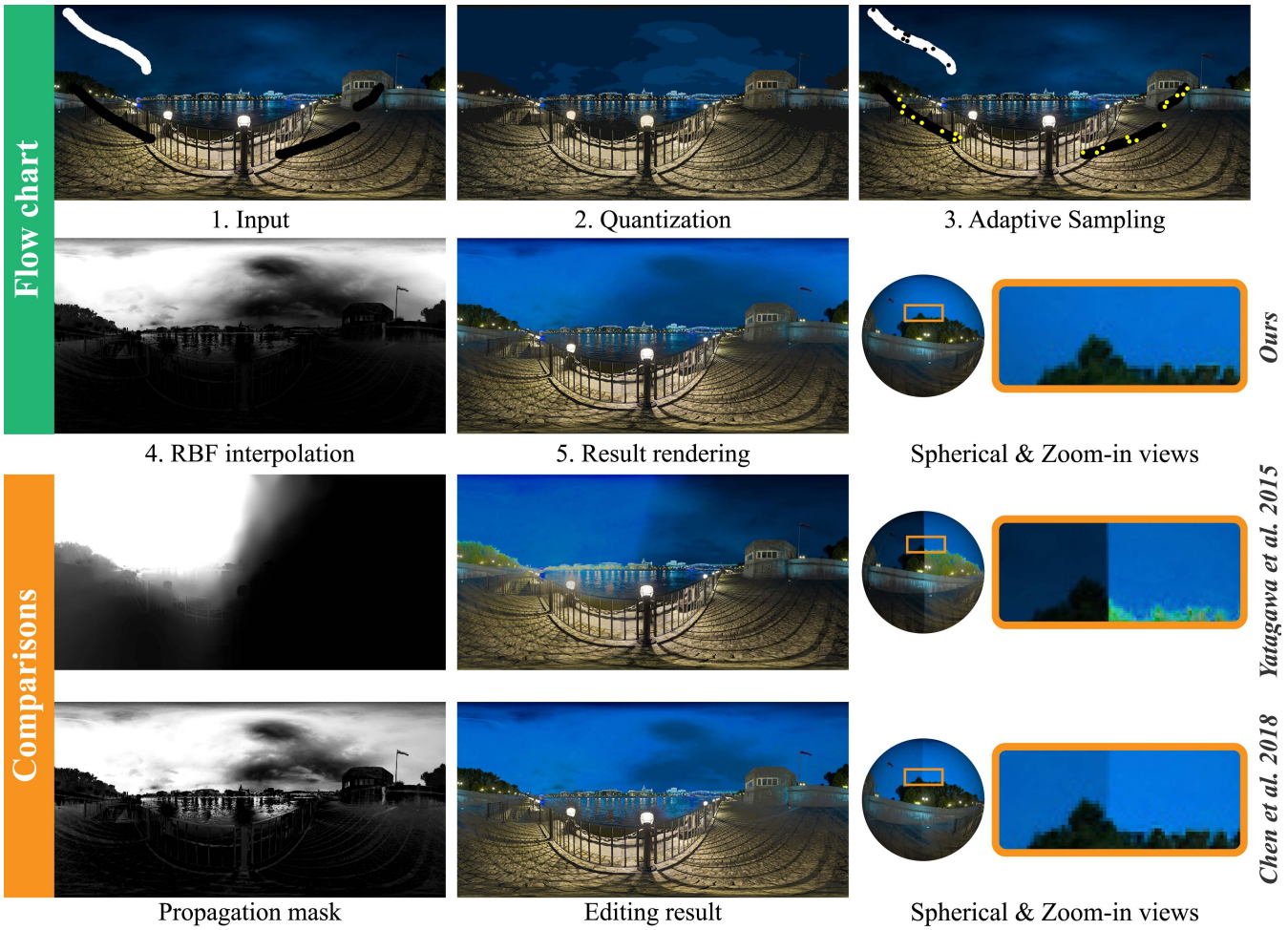


FIGURE 1. Flow chart and comparisons. We first quantize the input image and adaptively collect samples on strokes. Then, we perform RBF interpolation to obtain the edit at each pixel, which is visualized by the propagation mask. Finally, the editing result is rendered according to the calculated edits. We also compare our method with previous 2D edit propagation methods [10], [16], and provide spherical and zoom-in views for better visualization.

minimize the differences between user-specified edits and the edits by RBF interpolation:

$$\arg\min_{a_j} \sum_{k\in\Delta}(g_k - \sum_{j\in\Psi} a_j \sigma(\|\mathbf{f}_k - \mathbf{f}_j\|))^2, \tag{2}$$

where g_k refers to the user-specified edit at pixel k, and Δ is the set of all pixels on the user-specified strokes.

In our implementation, the energy function defined in Eq. 2 is constructed according to the selected samples and the user-specified edits on strokes. Then, the energy function is solved using the linear least squares with the non-negative coefficient constraints [7], and the edit at each pixel is calculated by Eq. 1. Finally, the editing result is rendered by applying the calculated edits (e.g., light, color) to each pixel.

B. ADAPTIVE SAMPLING

Li *et al.*'s proposal [7] of edit propagation by function interpolation is very promising, due to its efficiency and simplicity. In their method, the complexity of the edit propagation is independent of the number of pixels, and only related to the number of samples on strokes. However, their sampling strategy is not always effective when the strokes are dense, since the complexity of the function interpolation will increase dramatically under this condition. Besides, the randomly selected samples are difficult to represent the color distribution of strokes.

Inspired by [7], we propose an adaptive strategy for pixel sampling on strokes, which can greatly reduce the number of samples without sacrificing the visual quality of editing results. As described in Alg. 1, we first apply the K-means clustering [34] to quantize the colors of the input panoramic image P into P', which is visually similar to P but with fewer color levels, and can be used to analyze the color clusters on each stroke. Then, the number of samples on each stroke is set as:

$$N_u = \cos\xi_u \cdot |\Omega_u| \cdot \eta, \tag{3}$$

where $\xi_u \in [-90°, 90°]$, $|\Omega_u|$ is the number of clusters on each stroke, ξ_u is the average latitude of the area covered by the stroke, and η is a weight ranging from 1.0 to 2.0. In this

way, the uniform distribution of the samples on the sphere can be ensured.

Although Eq. 3 is effective to distribute samples to strokes, it cannot ensure the uniform distribution of samples in each cluster, due to the fact that the clusters contained on each stroke may have similar ξ_u and η values. For each stroke, we first enumerate the pixels in each color cluster, and then sort the clusters in a descending order by the number of pixels contained in each cluster. Note that clusters with too few pixels are discarded. In our implementation, we discard the clusters in which the number of pixels is less than 5% of the total number of pixels on the stroke. We assign samples to each cluster according to the number of pixels belonging to that cluster, and clusters with more pixels will be assigned with more samples. The number of samples in each cluster is calculated as:

$$N_{u,v} = \frac{|Q_{u,v}|}{\sum_v |Q_{u,v}|} N_u, \tag{4}$$

where $|Q_{u,v}|$ refers to the number of pixels belonging to the v^{th} cluster on the u^{th} stroke. Finally, we randomly select $N_{u,v}$ samples in cluster $\Omega_{u,v}$, and add the selected samples and user-specified edits to the adaptively selected sample set $\{\zeta_w\}$. In experiments, we found that Eq. 4 helps to distribute samples more uniformly, which is important for efficient and effective edit propagation.

Algorithm 1 Adaptive Sampling Algorithm

Input: Panoramic Image P and strokes $\{\gamma_u\}$
Output: Adaptively selected samples $\{\zeta_w\}$

```
P' = K-means_Quantization(P)
for each stroke γ_u in P' with color clusters Ω_u do
    Set the number of samples on each stroke by Eq. 3
    Sort the clusters in a descending order by the number
    of pixels contained in each cluster
    Discard the clusters with too few pixels
    for each cluster Ω_{u,v} do
        Calculate the number of samples in each cluster
        N_{u,v} by Eq. 4
        for m = 1 to N_{u,v} do
            Randomly select a sample in cluster Ω_{u,v}
            Add the sample and user-specified edit to {ζ_w}
        end for
    end for
end for
```

Fig. 2 shows the schematic diagram of sampling on a stroke of the quantized panorama. There are two different strategies for sampling. The random sampling cannot ensure the full coverage of all color clusters, while our adaptive sampling strategy can generate more representative samples.

C. EDIT PROPAGATION ON SPHERE

Edit propagation has been extensively studied in planar 2D images. However, simply extending it to 360° panoramas may introduce visible seams and inconsistent propagation. To solve these problems, Zhang *et al.* [6] proposed a manifold-preserving method for edit propagation on 360° panoramas. Although effective in many examples, their method still suffers from several problems, such as inaccurate spherical distance definition and complicated implementation.

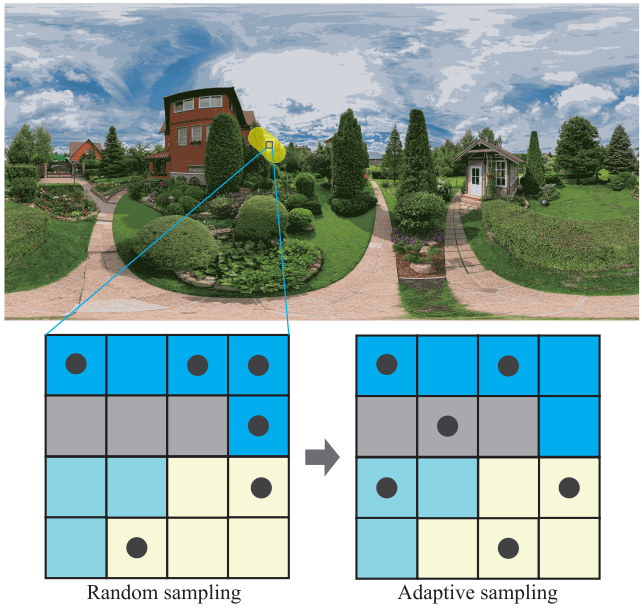


FIGURE 2. Schematic diagram of two different sampling strategies.

To effectively propagate sparse edits on 360° panoramas, we redefine the function interpolation based on spherical coordinates. See Fig. 3, for a pixel $S(\alpha, \beta)$ on an equirectangular panorama, we first map it to the spherical coordinate system. More specifically, each planar pixel (α, β) is projected onto a unit sphere where their 3D coordinates $S'(x, y, z)$ can be computed as:

$$\begin{cases} x = \cos\theta \cos\phi \\ y = \cos\theta \sin\phi \\ z = \sin\theta. \end{cases} \tag{5}$$

Here θ and ϕ refer to longitude and latitude respectively, and can be calculated as:

$$\begin{cases} \theta = 2\pi(\alpha - W/2.0)/W \\ \phi = -\pi(\beta - H/2.0)/H, \end{cases} \tag{6}$$

where W and H refer to the width and height of the input equirectangular panorama respectively.

The main difference between spherical and planar edit propagation lies in the spatial distance measurement. Different from [6], which uses the Euclidean distance to approximate the actual distance on the sphere, we use the great circle distance to measure the shortest distance between pixels on the surface of a unit sphere. As illustrated in Fig. 3, the spherical distance between S' and T' is actually the arc length

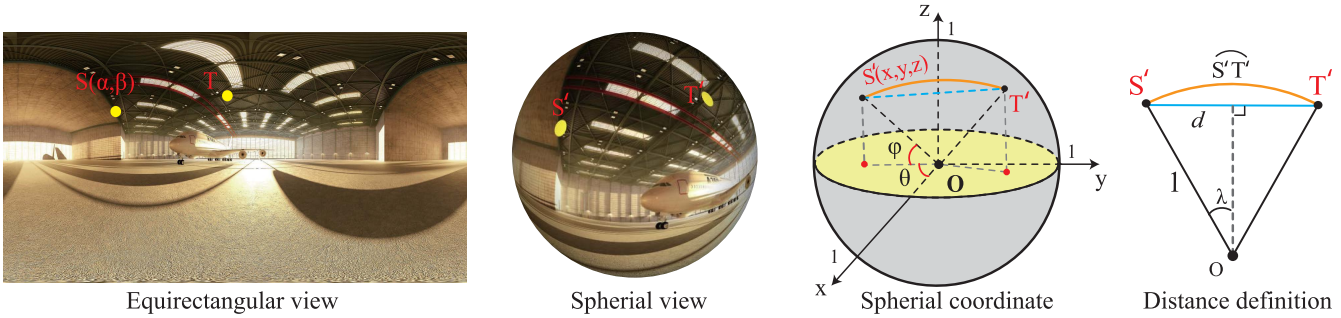


FIGURE 3. Pixel definition on the spherical coordinate system, and the spherical distance definition.

$\widehat{S'T'}$ corresponding to the chord length $\overline{S'T'}$ in the unit circle, and the distance of $\widehat{S'T'}$ can be calculated as:

$$|\widehat{S'T'}| = 2\lambda = 2\arcsin(|\overline{S'T'}|/2), \tag{7}$$

where $|\overline{S'T'}|$ refers to the Euclidean distance between S' and T' on sphere.

After defining the spherical distance, the distance between feature vectors defined in Eqs. 1 and 2 can be rewritten as:

$$\sqrt{\|\mathbf{c}_{S'}/\sigma_c - \mathbf{c}_{T'}/\sigma_c\|^2} + |\widehat{S'T'}|, \tag{8}$$

where $\mathbf{c}_{S'}$ and $\mathbf{c}_{T'}$ refer to the color vectors of the points S' and T' after the spherical projection.

Fig. 1 gives a comparison between typical 2D edit propagation methods [10], [16] and our method. Results show that our method can effectively remove the seam on the borders and smoothly propagate features near the top and bottom regions due to the spherical definition.

D. ACCELERATION

Since 360° panoramas capture the entire scene in all directions, they typically contain much more pixels than ordinary images, usually in the millions or more, which brings much more computational burden for editing tasks. For efficient propagation, we adopted the multi-resolution speedup strategy in [6], which performed edit propagation in the downsampled input and then upsampled the result to the original size using the guided image filtering [35]. To propagate the edits on sphere, we use the great circle distance to measure the actual distance between pixels, see Eqs. 7 and 8. However, the calculation of **arcsin** and **sqrt** are very time-consuming. We instead pre-build a look-up table for both functions ahead, and the function value of a given point can be queried from the look-up table or approximated by its neighbors using linear interpolation.

IV. RESULTS

A. EVALUATIONS

The images used in the experiments are collected from the Internet by the authors, and all of them have free license. Our experiments are performed on a PC with an AMD Ryzen 7 Pro 4750U, 1.7GHz CPU, and 32GB RAM. We implemented our algorithm in C++, and set $K = 30$, $\sigma_c = 0.1$, $\sigma_p = 1.0$ for all cases. The evaluation of the algorithm includes the comparison with previous state-of-the-art methods in quality and in efficiency. In all examples, the *white* strokes are used to indicate regions to relight, and the *black* ones are to indicate regions to remain unchanged, while *other colors* indicate the target color of the editing tasks. To demonstrate the effectiveness of our method, we apply our method to several different editing tasks, including recoloring, relighting, alpha blending, cartoon editing, and video editing. Comparisons are made with state-of-the-art methods on these tasks, and we use propagation masks, spherical and zoom-in views for the visualization and comparison of results.

Compared with previous methods, our method is more efficient and effective on 360° panoramas, especially for those with a huge number of pixels. The reason is that previous methods did not consider the spherical characteristic of 360° panoramas, and the feature vector and distance metric are defined on 2D plane, which may lead to a visible "seam" on the left and right borders, and inconsistent propagations near the top and bottom regions. In addition, previous methods mainly focus on the editing of normal size images, and they are less effective and efficient when editing panoramas with tens of millions of pixels.

We first compare our method with previous sampling-based method [16], and the results in Fig. 1 show that their method is not effective in propagating edits over the sphere. The reason is that they did not consider the spherical structure while sampling, and their parameters need to be carefully adjusted, which limits its practicability. In addition, their method is very time-consuming, and requires 5∼20 seconds to process images with normal size. In contrast, our method is designed according to the spherical nature of 360° panoramas. We improve the sampling-based method by applying adaptive sampling on sphere, which can produce seam-free and smooth editing results with fewer samples. In addition, our method can generate visually satisfying results without too much careful tuning of parameters. More comparisons with [10] also show the advantages of our method, see the spherical and zoom-in views.

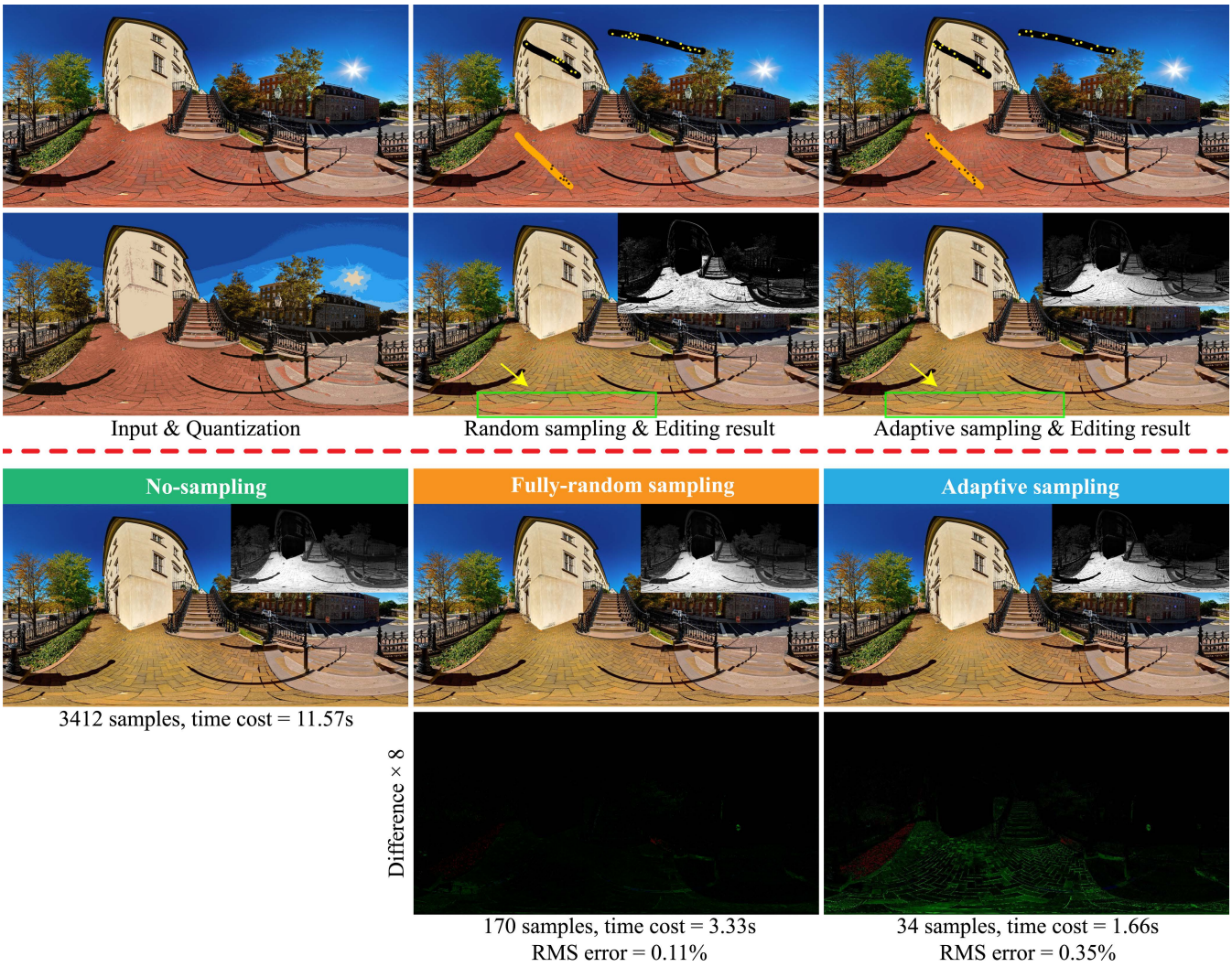


FIGURE 4. Comparison of different sampling strategies. In the upper part, column 1 shows the input panorama and its quantization result, and columns 2-3 show the editing results by using random sampling and our adaptive sampling strategies. For better comparison, we use the marked arrows to indicate the regions where two different approaches generate most different results. Samples and propagation masks are attached on the image for better illustration. In the lower part, we further compare no-sampling, fully-random sampling [7] and our adaptive sampling strategies in terms of the number of samples, accuracy, and computation time, and provide the images showing the differences (multiplied by 8) between the results generated by the sampling and non-sampling methods.

TABLE 1. Performance comparison of our method with sampling-based method [7] and state-of-the-art spherical edit propagation [6].

	Fig. 4	Fig. 5	Fig. 6	Fig. 7	Fig. 9(1)	Fig. 9(2)	Fig. 12(1)
Type	image	image	image	image	image	image	video(**150** frames)
Resolution	4800 × 2400	2048 × 1024	5000 × 3500	1600 × 800	6144 × 3072	8000 × 4000	1366 × 684
Li et al. [7]	3.33s	3.13s	5.13s	1.83s	3.57s	7.31s	51.7s
Zhang et al. [6]	3.93s	2.04s	4.17s	2.11s	3.84s	6.22s	49.6s
Our (Total)	**1.66**s	**1.34**s	**2.44**s	**1.03**s	**2.37**s	**4.06**s	**39.62**s
Adaptive Sampling	0.23s	0.27s	0.31s	0.33s	0.53s	0.49s	0.47s
RBF interpolation	0.72s	0.92s	1.08s	0.49s	0.81s	1.95s	29.21s
Multi-resolution	0.71s	0.15s	1.05s	0.21s	1.03s	1.62s	9.94s

Fig. 4 shows the comparison of different sampling strategies. We first compare the random sampling strategy [7] and our adaptive sampling strategy. For the sake of fairness, we set the same number of samples for both methods. In this experiment, the input image was quantized using 30 clusters, and 34 samples were extracted from the strokes. The experimental results and comparison show that our adaptive strategy can generate more uniformly distributed samples to represent

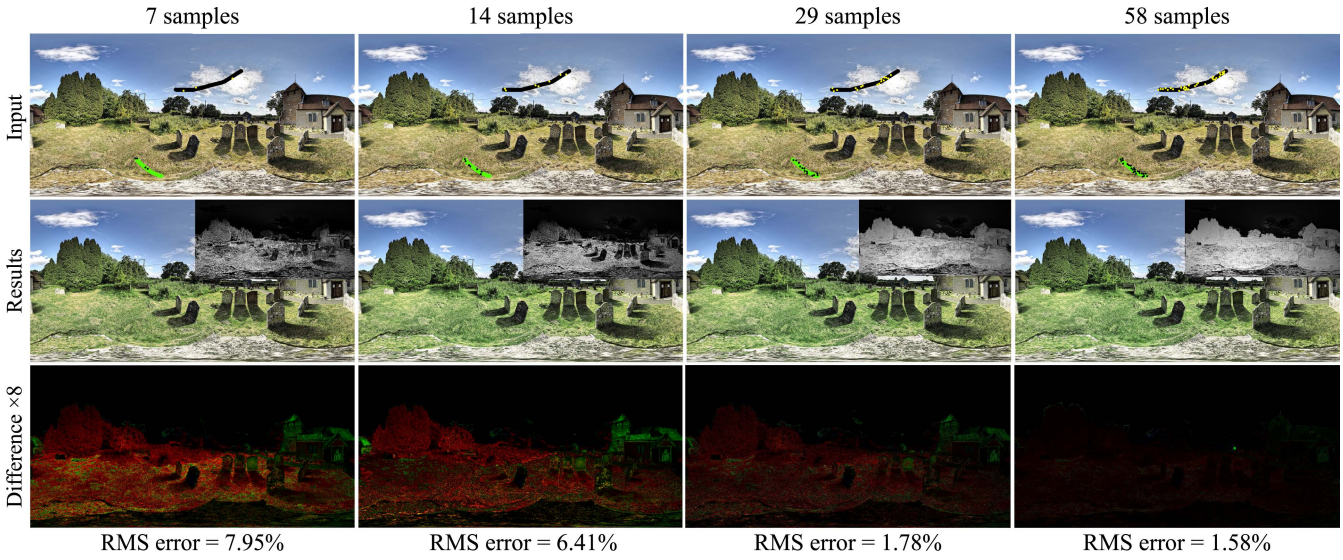


FIGURE 5. Edit propagation results using different number of samples as RBF centers. Columns 1 to 4 show the editing results of our method and the difference images (multiplied by 8 times) to the results of [7], using 7, 14, 29 and 58 samples respectively. We also provide propagation masks for better comparison.

FIGURE 6. Results of step-by-step editing. Column 1 shows the strokes and result by the first step editing, which turns the grass yellow. Column 2 shows the strokes and result by the second step editing, which darkens the sky. Column 3 shows the original input strokes, and the final result that combines the recoloring and relighting results. We provide a propagation mask at each step, and a spherical view of the final result.

the color features on strokes, and can generate more satisfying editing results. We also compare no-sampling, fully-random sampling [7] and our adaptive sampling strategies in terms of the number of samples, accuracy, and computation time. In our implementation, the no-sampling method takes all pixels in the input strokes as samples, and the fully-random sampling [7] randomly selects 5% of all pixels in the input strokes as samples. Results show that more samples can achieve more accurate propagation, but take much more time. The differences between the results generated by the sampling and no-sampling methods, and the relative mean square (RMS) errors vividly show that our method can efficiently produce comparable results using fewer samples.

To evaluate the effects of using different number of samples, we compare the results of edit propagation using different number of samples on strokes. See Fig. 5, in this example, we set $\xi_u = -72.2°$ for the green stroke, $\xi_u = 36.2°$ for the black one, $\eta = 1.6$ for all strokes, and the total number of samples is 29. We also make comparisons by increasing or decreasing the number of samples, and the number of samples is set manually. The editing results and propagation masks vividly show that too few samples can not effectively propagate features to the whole image, and adding samples can improve the edit propagation. However, when the number of samples reaches a certain amount, it will not bring significant improvement. For quantitative evaluations, we also compare our results generated from a different

FIGURE 7. Comparison with state-of-the-art methods. The input panorama and strokes are provided in the first line. Based on the above input, we provide editing results by [6], [9] and our method. We also provide propagation masks, spherical and zoom-in views for better comparison.

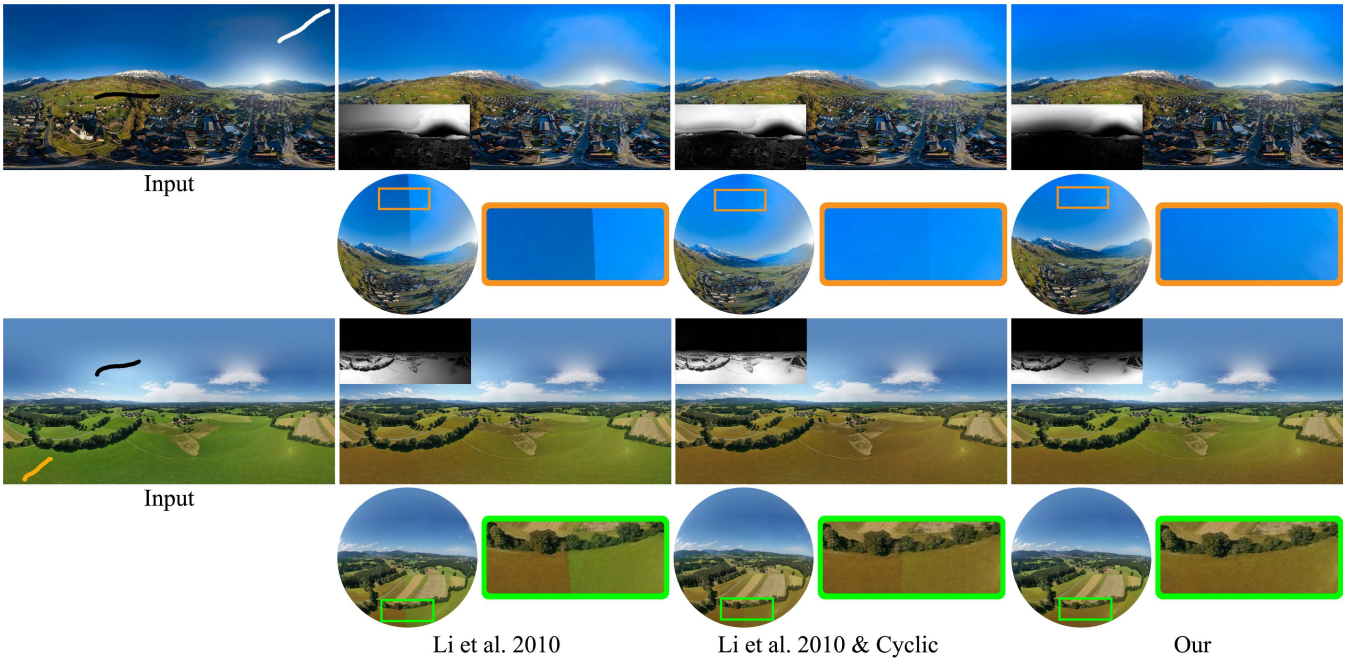


FIGURE 8. Comparison with edit propagation method on cyclic images. We give two sets of results, and each set gives the result by Li *et al.* [7], their improved result by considering cyclic images, and our result. We provide propagation masks, spherical and zoom-in views for better comparison.

number of samples with the result of [7], using the difference images and the RMS errors, which show that our method can achieve comparable results with fewer samples. Actually, for 360° panoramas with tens of millions of pixels, the large number of samples may introduce a considerable increase in computation. See Tab. 1, the method in [7] takes much more time than our method due to the large number of randomly selected samples.

To show the process of editing by multiple strokes, we provide an example of edit propagation which targets both recoloring and relighting, see Fig. 6. After drawing multiple strokes (see the last column), we first focus on the

FIGURE 9. More results and comparisons. We provide more examples and comparisons with Chen *et al.* [10]. The spherical and zoom-in views vividly show the advantages of our method.

FIGURE 10. Foreground selection and alpha blending of 360° panoramas.

recoloring, so the color of strokes for relighting turns black, which indicates that the sky region should keep unchanged. Then, we move to the sky relighting, and the yellow strokes for recoloring turn black, which helps to keep the grassland unchanged. The final editing result is obtained by a linear combination of the results of the above two steps.

Comparison with state-of-the-art methods is shown in Fig. 7. The first column shows the results produced by the manifold preserving method in [9], which generates visible seams due to the limitation of 2D planar distance metrics. As shown in the middle column, the state-of-the-art work for spherical edit propagation [6] can effectively remove the seam on borders and produce more visually pleasing results, due to its spherical distance measurement. However, since the manifold preserving structure defined on sphere is not stable, their method still generates artifacts in smooth regions, see the red arrows in the sky region. Compared with [6], our spherical edit propagation based on function interpolation generates more natural and smooth results.

We further compare our results with the edit propagation method on cyclic images, which can be obtained by projecting the equirectangular panorama onto a cylinder. See Fig. 8, compared with the traditional 2D edit propagation method [7], the cylinder-based propagation can effectively alleviate the seam issue on the left and right borders of equirectangular images due to the distance metrics defined on a cylinder. However, disconnections can still be seen from the zoom-in views. In contrast, our sphere-based propagation can completely remove the seam on borders and enable smoother propagation of edits on 360° panoramas.

We give more results in Fig. 9, where comparisons are made between [10] and our method. The spherical and

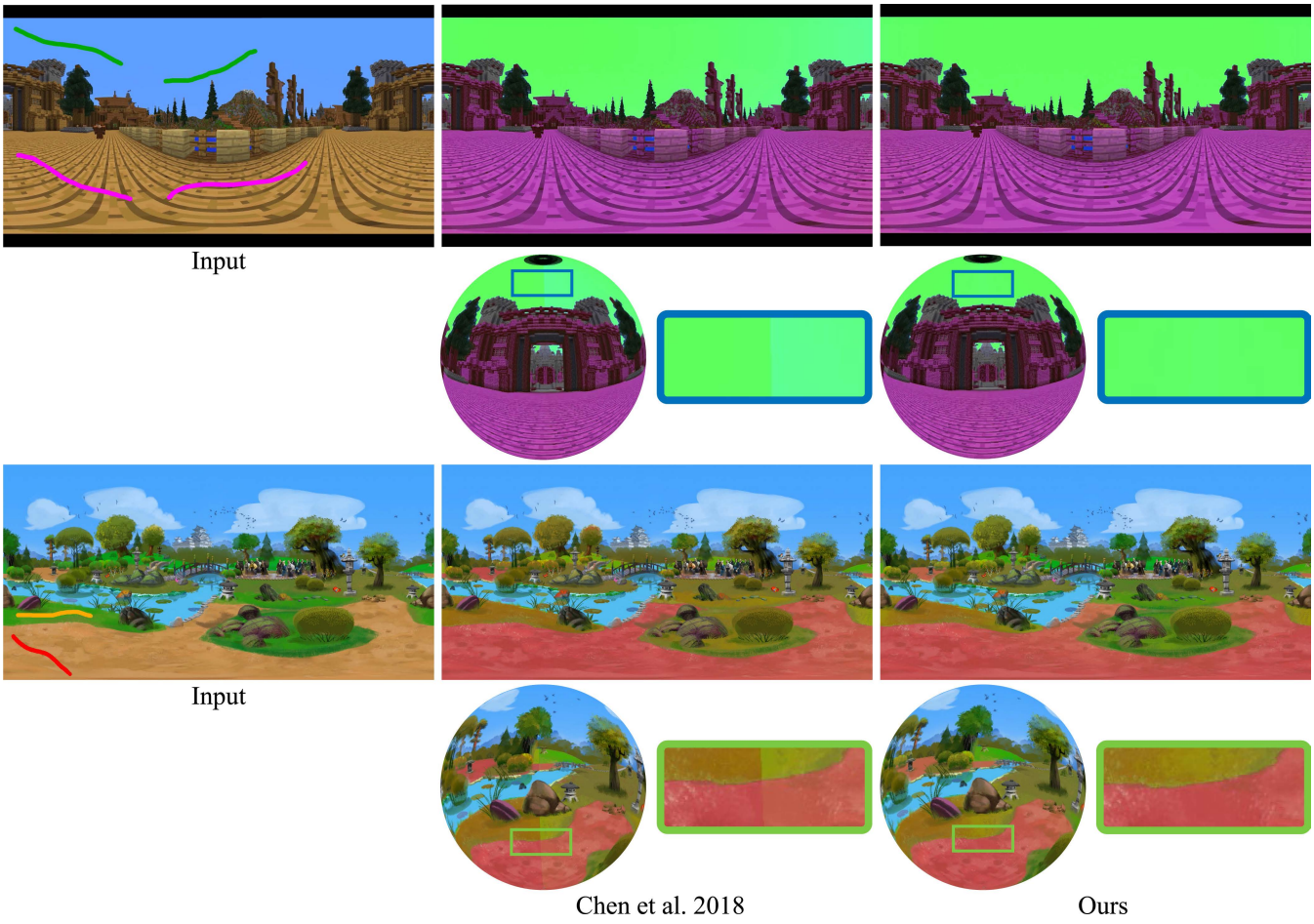


FIGURE 11. Edit propagation of 360° cartoon panoramas.

zoom-in views vividly show the advantages of our method in these examples, where the edits are smoothly propagated, and the left and right borders remain continuous after editing.

B. PERFORMANCE

We report the performance of different methods in Tab. 1, including the 2D edit propagation method based on the RBF interpolation [7], the state-of-the-art spherical manifold preserving method [6], and our method. Take Fig. 4 for example, our total time cost is 1.66*s*, including the adaptive sampling, RBF interpolation, and multi-resolution processing. In comparison, the traditional RBF interpolation based method [7] takes much more time due to the random sampling and the large number of samples needed to achieve appropriate results. The method of Zhang *et al.* [6] is also time-consuming, which is caused by the K-D tree manipulations and energy minimization. Experiments on multiple cases show that our method is much more efficient than those state-of-the-art methods.

C. APPLICATIONS

Our method also enables many interesting applications in 360° panorama editing. Fig. 10 shows examples of alpha blending of 360° panoramas. We first let users draw white and black strokes to specify foreground and background respectively, and the propagation mask is calculated for further editing. We then blend the input and target 360° panoramas using the propagation masks as the alpha values. The two examples in Fig. 10 show that the ground and sky of the input panoramas are seamlessly replaced by the grassland and blue sky of other panoramas. We also apply our edit propagation to cartoon panoramas with hard boundaries. See Fig. 11, results and comparison with [10] vividly show the effectiveness and advantages of our method.

We further extend our method to 360° video editing. To propagate edits in videos, another time dimension should be added, so the feature vector becomes $\mathbf{f}_i = \{\mathbf{c}_i/\sigma_c, \mathbf{p}_i/\sigma_p, t_i/\sigma_t\}$, where σ_t is used to control the importance of the time dimension. To effectively propagate edits across frames, we set a big value 10 to σ_t, and t_i is normalized as $t_i = t_i/N_f$, where N_f is the number of frames. The feature vector distance in Eq. 8 is modified by adding the temporal distance $|t_i/\sigma_t - t_j/\sigma_t|$. Fig. 12 gives the editing results on 360° videos, and the edits on key frames can be effectively propagated across other frames. We also provide

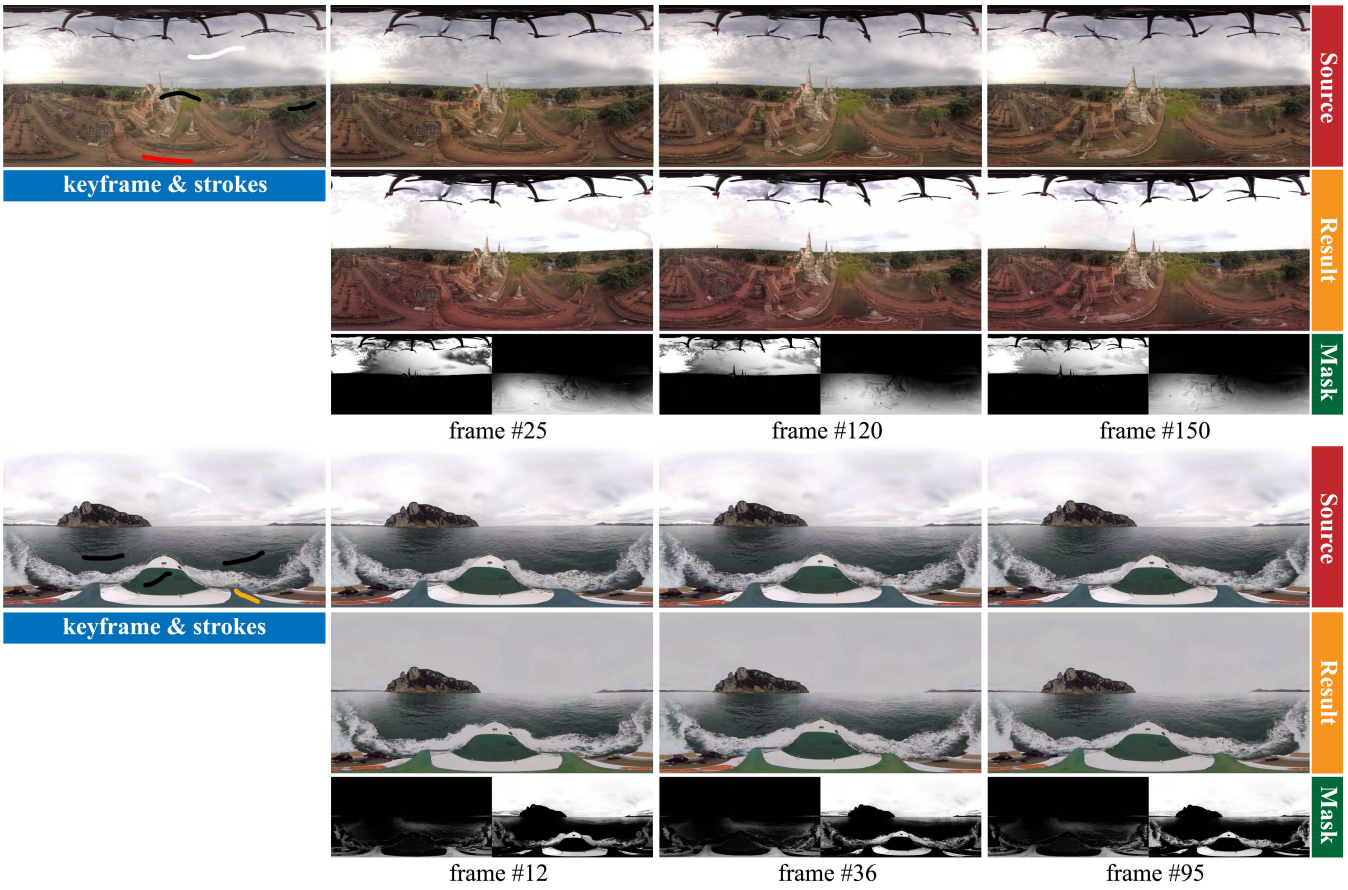


FIGURE 12. Edit propagation on 360° videos. To evaluate the propagation results, we also provide the propagation masks of different strokes.

the propagation masks of different strokes to evaluate the propagation result of each frame.

V. CONCLUSION

This paper proposes a function interpolation based method for fast and effective edit propagation on 360° panoramas. We first quantize the colors of the input panorama and then adaptively select representative samples to construct smooth functions in the affinity space, according to the diversity of local color distribution. For seam-free and consistent propagation on the sphere, we incorporate the spherical metrics into the calculation of pixel distance on equirectangular panoramas. We further propose an acceleration scheme for fast propagation on ultra high definition panoramas, including the multi-resolution strategy and the look-up table based function approximation. Experiments and comparisons on multiple editing tasks demonstrate that our method is effective for 360° panoramas editing and is advantageous over existing state-of-the-art methods in terms of speed and visual effects.

Our method still suffers from some limitations: (1) the function interpolation based method may produce halo artifacts at regions with color blending; (2) since there are no hard constraints, the results might be unsatisfactory when placing different edits on regions with similar colors.

In the future, we will add more semantic features such as scene structure [36], emotion [37], and attention mechanism [38], and apply the deep learning framework [39] to enable high-level edits on 360° panoramas.

REFERENCES

[1] X. Li, R. Feng, X. Guan, H. Shen, and L. Zhang, "Remote sensing image mosaicking: Achievements and challenges," *IEEE Geosci. Remote Sens. Mag.*, vol. 7, no. 4, pp. 8–22, Apr. 2019.

[2] X. Wei, W. Yan, Q. Zheng, M. Gu, K. Su, G. Yue, and Y. Liu, "Image redundancy filtering for panorama stitching," *IEEE Access*, vol. 8, pp. 209113–209126, 2020.

[3] A. Levin, D. Lischinski, and Y. Weiss, "Colorization using optimization," *ACM Trans. Graph.*, vol. 23, no. 3, pp. 689–694, 2004.

[4] D. Lischinski, Z. Farbman, M. Uyttendaele, and R. Szeliski, "Interactive local adjustment of tonal values," *ACM Trans. Graph.*, vol. 25, no. 3, pp. 646–653, 2006.

[5] X. An and F. Pellacini, "Appprop: All-pairs appearance-space edit propagation," *ACM Trans. Graph.*, vol. 27, no. 3, p. 40, 2008.

[6] Y. Zhang, F.-L. Zhang, Y.-K. Lai, and Z. Zhu, "Efficient propagation of sparse edits on 360° panoramas," *Comput. Graph.*, vol. 96, pp. 61–70, May 2021.

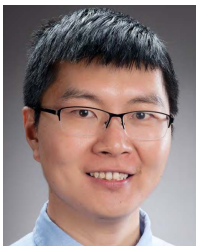
[7] Y. Li, T. Ju, and S.-M. Hu, "Instant propagation of sparse edits on images and videos," *Comput. Graph. Forum*, vol. 29, no. 7, pp. 2049–2054, 2010.

[8] X. Chen, D. Zou, Q. Zhao, and P. Tan, "Manifold preserving edit propagation," *ACM Trans. Graph.*, vol. 31, no. 6, pp. 132:1–132:7, 2012.

[9] L.-Q. Ma and K. Xu, "Efficient manifold preserving edit propagation with adaptive neighborhood size," *Comput. Graph.*, vol. 38, pp. 167–173, Feb. 2014.

[10] Y. Chen, G. Zong, G. Cao, and J. Dong, "Efficient manifold-preserving edit propagation using quad-tree data structures," *Multimedia Tools Appl.*, vol. 77, no. 6, pp. 6699–6712, Mar. 2018.
[11] K. Xu, Y. Li, T. Ju, S. Hu, and T. Liu, "Efficient affinity-based edit propagation using K-D tree," *ACM Trans. Graph.*, vol. 28, no. 5, p. 118, 2009.
[12] K. He, C. Rhemann, C. Rother, X. Tang, and J. Sun, "A global sampling method for alpha matting," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Colorado Springs, CO, USA, Jun. 2011, pp. 2049–2056.
[13] G. Yao, Z. Zhao, and S. Liu, "A comprehensive survey on sampling-based image matting," in *Comp. Graph. Forum*, vol. 36 no. 6, pp. 613–628, Dec. 2017.
[14] H. Huang, Y. Liang, X. Yang, and Z. Hao, "Pixel-level discrete multiobjective sampling for image matting," *IEEE Trans. Image Process.*, vol. 28, no. 8, pp. 3739–3751, Aug. 2019.
[15] X. Bie, H. Huang, and W. Wang, "Real time edit propagation by efficient sampling," *Comput. Graph. Forum*, vol. 30, no. 7, pp. 2041–2048, Sep. 2011.
[16] T. Yatagawa and Y. Yamaguchi, "Sparse pixel sampling for appearance edit propagation," *Vis. Comput.*, vol. 31, nos. 6–8, pp. 1101–1111, Jun. 2015.
[17] L.-Q. Ma and K. Xu, "Efficient antialiased edit propagation for images and videos," *Comput. Graph.*, vol. 36, no. 8, pp. 1005–1012, Dec. 2012.
[18] C. Oh, S. Ryu, Y. Kim, J. Kim, T. Park, and K. Sohn, "Sparse edit propagation for high resolution image using support vector machines," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Quebec City, QC, Canada, Sep. 2015, pp. 4042–4046.
[19] X. Chen, J. Li, D. Zou, and Q. Zhao, "Learn sparse dictionaries for edit propagation," *IEEE Trans. Image Process.*, vol. 25, no. 4, pp. 1688–1698, Apr. 2016.
[20] Y. Endo, S. Iizuka, Y. Kanamori, and J. Mitani, "DeepProp: Extracting deep features from a single image for edit propagation," *Comput. Graph. Forum*, vol. 35, no. 2, pp. 189–201, 2016.
[21] Y. Gui and G. Zeng, "Joint learning of visual and spatial features for edit propagation from a single image," *Vis. Comput.*, vol. 36, no. 3, pp. 469–482, Mar. 2020.
[22] M. Xu, C. Li, S. Zhang, and P. L. Callet, "State-of-the-art in 360° video/image processing: Perception, assessment and compression," *IEEE J. Sel. Topics Signal Process.*, vol. 14, no. 1, pp. 5–26, Jan. 2020.
[23] M. Wang, X.-Q. Lyu, Y.-J. Li, and F.-L. Zhang, "VR content creation and exploration with deep learning: A survey," *Comput. Vis. Media*, vol. 6, no. 1, pp. 3–28, Mar. 2020.
[24] J. Jung, B. Kim, J. Y. Lee, B. Kim, and S. Lee, "Robust upright adjustment of 360 spherical panoramas," *Vis. Comput.*, vol. 33, nos. 6–8, pp. 737–747, 2017.
[25] R. Jung, A. S. J. Lee, A. Ashtari, and J. Bazin, "Deep360up: A deep learning-based approach for automatic VR image upright adjustment," in *Proc. IEEE Conf. Virtual Reality 3D User Interfaces (VR)*, Osaka, Japan, Mar. 2019, pp. 1–8.
[26] J. Kopf, "360° video stabilization," *ACM Trans. Graph.*, vol. 35, no. 6, pp. 195:1–195:9, 2016.
[27] C. Tang, O. Wang, F. Liu, and P. Tan, "Joint stabilization and direction of 360° videos," *ACM Trans. Graph.*, vol. 38, no. 2, pp. 18:1–18:13, 2019.
[28] K. Kang and S. Cho, "Interactive and automatic navigation for 360° video playback," *ACM Trans. Graph.*, vol. 38, no. 4, pp. 108:1–108:11, 2019.
[29] M. Wang, Y.-J. Li, W.-X. Zhang, C. Richardt, and S.-M. Hu, "Transitioning360: Content-aware NFoV virtual camera paths for 360° video playback," in *Proc. IEEE Int. Symp. Mixed Augmented Reality (ISMAR)*, Recife/Porto de Galinhas, Brazil, Nov. 2020, pp. 185–194.
[30] W.-S. Lai, Y. Huang, N. Joshi, C. Buehler, M.-H. Yang, and S. B. Kang, "Semantic-driven generation of hyperlapse from 360° video," *IEEE Trans. Vis. Comput. Graphics*, vol. 24, no. 9, pp. 2610–2621, Sep. 2018.
[31] Z. Zhu, R. R. Martin, and S.-M. Hu, "Panorama completion for street views," *Comput. Vis. Media*, vol. 1, no. 1, pp. 49–57, Mar. 2015.
[32] Q. Zhao, L. Wan, W. Feng, J. Zhang, and T.-T. Wong, "360 panorama cloning on sphere," 2017, *arXiv:1709.01638*.
[33] J. S. Sumantri and I. K. Park, "360 panorama synthesis from a sparse set of images on a low-power device," *IEEE Trans. Comput. Imag.*, vol. 6, pp. 1179–1193, 2020.
[34] J. Macqueen, "Some methods for classification and analysis of multivariate observations," in *Proc. 5th Berkeley Symp. Math. Statist. Probab.*, 1967, pp. 281–297.
[35] K. He, J. Sun, and X. Tang, "Guided image filtering," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 6, pp. 1397–1409, Jun. 2013.
[36] M. Wang, X.-N. Fang, G.-W. Yang, A. Shamir, and S.-M. Hu, "Prominent structures for video analysis and editing," *IEEE Trans. Vis. Comput. Graphics*, vol. 27, no. 7, pp. 3305–3317, Jul. 2021.
[37] J. Yang, J. Li, X. Wang, Y. Ding, and X. Gao, "Stimuli-aware visual emotion analysis," *IEEE Trans. Image Process.*, vol. 30, pp. 7432–7445, 2021.
[38] G. Ma, S. Li, C. Chen, A. Hao, and H. Qin, "Stage-wise salient object detection in 360° omnidirectional image via object-level semantical saliency ranking," *IEEE Trans. Vis. Comput. Graphics*, vol. 26, no. 12, pp. 3535–3545, Dec. 2020.
[39] O. Perel, O. Anschel, O. Ben-Eliezer, S. Mazor, and H. Averbuch-Elor, "Learning multimodal affinities for textual editing in images," *ACM Trans. Graph.*, vol. 40, no. 3, pp. 26:1–26:16, 2021.

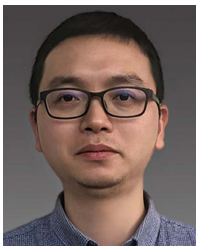
YUN ZHANG received the bachelor's and master's degrees from Hangzhou Dianzi University, China, in 2006 and 2009, respectively, and the Ph.D. degree from Zhejiang University, in 2013. From February 2018 to August 2018, he was a Visiting Scholar at Cardiff University. He is currently an Associate Professor at the Communication University of Zhejiang. His research interests include computer graphics, image and video editing, and computer vision. He is a member of CCF.



FANG-LUE ZHANG (Member, IEEE) received the Ph.D. degree from Tsinghua University, in 2015. He is currently a Lecturer with the Victoria University of Wellington, Wellington, New Zealand. His research interests include image and video editing, computer vision, and computer graphics. He is a member of ACM. He received the Victoria Early-Career Research Excellence Award, in 2019, and the Fast-Start Marsden Grant from the New Zealand Royal Society, in 2020.

ZHE ZHU received the bachelor's degree from Wuhan University, in 2011, and the Ph.D. degree from the Department of Computer Science and Technology, Tsinghua University, in 2017. He was a Senior Research Associate at Duke University. He is currently a Senior Software Engineer at Mathworks. His research interests include computer vision and computer graphics.

LIDONG WANG was born in 1982. She received the Ph.D. degree from the College of Computer Science and Technology, Zhejiang University. She is currently an Associate Professor at Hangzhou Normal University. Her current research interests include image processing, computer graphics, machine learning, and text mining.



YAO JIN received the B.S. degree in apparel engineering and the M.S. degree in computer science from Zhejiang Sci-Tech University, China, in 2007 and 2010, respectively, and the Ph.D. degree from Zhejiang University, China, in 2015. He is currently an Associate Professor with the Department of Digital Media Technology, Zhejiang Sci-Tech University. His research interests include computer graphics and digital geometry process.